PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2001-244820
(43)Date of publication of application : 07.09.2001

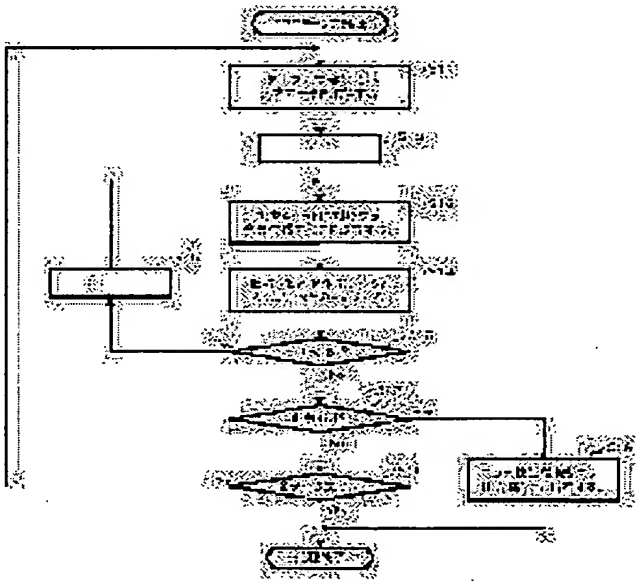
(51)Int.Cl. H03M 13/09
G06F 11/10
G06F 12/16
H04L 1/00

(21)Application number : 2000-054468
(22)Date of filing : 29.02.2000

(71)Applicant : KENWOOD CORP
(72)Inventor : HIGUCHI TOMOHISA

(54) ERROR DETECTING DEVICE, CRC CHECK METHOD, AND RECORDING MEDIUM

(57)Abstract:
PROBLEM TO BE SOLVED: To provide an error detecting device which can make a CRC check at a high speed.
SOLUTION: An arithmetic procession once obtaining object data from a data input part (step S11) specifies four object bits in the object data and obtains summed data corresponding to the specified four object bits from a generation data table (step S13). The arithmetic procession performs exclusive OR operation between the obtained summed data and input data (step S14). The arithmetic procession repeats the processes of the steps S13 and S14 six times. The arithmetic procession performs exclusive OR operation between the object data and summed data of the six times to decide whether or not residue is generated (step S17). The arithmetic procession when deciding that a residue is generated generates and supplies error detection signal to an error output part (step S18).



LEGAL STATUS

[Date of request for examination] 12.04.2002

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

Copyright (C); 1998,2003 Japan Patent Office

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号 ✓
特開2001－244820
（P2001－244820A）
(43)公開日　平成13年９月７日(2001.9.7)

(51)Int.Cl.⁷		識別記号	FI		テーマコード(参考)
Ｈ０３Ｍ	13/09		Ｈ０３Ｍ 13/09		５Ｂ００１
Ｇ０６Ｆ	11/10	３３０	Ｇ０６Ｆ 11/10	３３０Ａ	５Ｂ０１８
	12/16	３２０	12/16	３２０Ｂ	５Ｊ０６５
Ｈ０４Ｌ	1/00		Ｈ０４Ｌ 1/00	Ａ	５Ｋ０１４

審査請求　未請求　請求項の数５　ＯＬ　（全 13 頁）

(21)出願番号　特願2000－54468(P2000－54468)

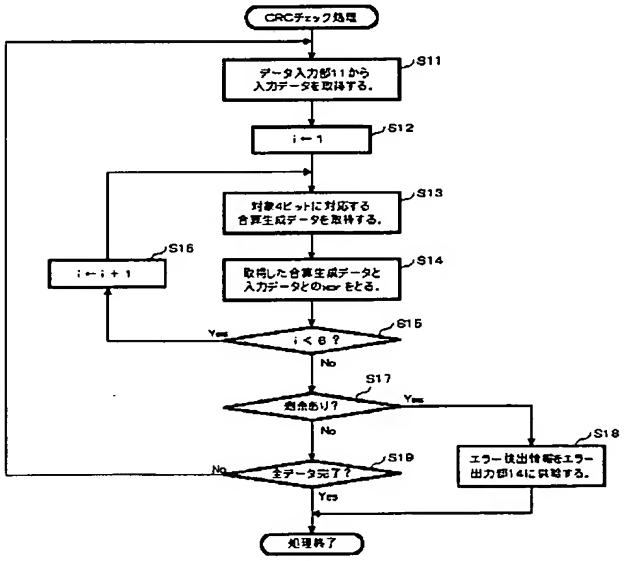
(22)出願日　平成12年２月29日(2000.2.29)

(71)出願人　000003595
株式会社ケンウッド
東京都渋谷区道玄坂１丁目14番６号
(72)発明者　樋口　知久
東京都渋谷区道玄坂１丁目14番６号　株式会社ケンウッド内
(74)代理人　100077850
弁理士　芦田　哲仁朗　（外１名）

Ｆターム(参考)　5B001 AA04 AB01 AC01 AD03 AD06 AD07
5B018 GA01 HA11
5J065 AB01 AC03 AD08 AE01 AF01 AG01 AH04 AH15 AH18 AH19
5K014 BA06 EA01 EA02

(54)【発明の名称】　誤り検出装置、ＣＲＣチェック方法及び記録媒体

(57)【要約】
【課題】　ＣＲＣチェックを高速に行うことのできる誤り検出装置を提供することである。
【解決手段】　演算処理部は、データ入力部から対象データを取得すると（ステップＳ１１）、対象データ中の対象４ビットを特定し、特定した対象４ビットに対応する合算生成データを生成データテーブルから取得する（ステップＳ１３）。演算処理部は、取得した合算生成データと入力データとの排他的論理和を演算する（ステップＳ１４）。演算処理部は、このステップＳ１３、Ｓ１４の処理を６回繰り返す。演算処理部は、６回にわたる対象データと合算生成データとの排他的論理和の演算により、剰余が発生しているか否かを判別する（ステップＳ１７）。演算処理部は、剰余が発生していると判別した場合、エラー検出情報を生成して、エラー出力部に供給する（ステップＳ１８）。



1

【特許請求の範囲】

【請求項１】複数ビット分のビット配列に従って、所定の生成多項式が合算されて生成された合算生成データを予め記憶する記憶手段と、

ビット列同士の排他的論理和を演算する演算手段と、

ＣＲＣ（Cyclic Redundancy Check）チェック対象のデータを入力する入力手段と、

前記入力手段が入力したデータ又は、前記演算手段により前回演算されて生じた演算結果データから複数ビット分のビット配列を特定し、特定したビット配列に対応する合算生成データを前記記憶手段から取得する取得手段と、

前記入力手段が入力したデータ又は、前記演算手段により前回演算されて生じた演算結果データと、前記取得手段が取得した合算生成データとの演算を前記演算手段に所定回数行わせる制御手段と、

前記制御手段の制御により前記演算手段にて所定回数繰り返された演算により生じた演算結果に従って、前記入力手段が入力したデータの誤りを検出するエラー検出手段と、

を備えることを特徴とする誤り検出装置。

【請求項２】１バイト分のビット配列に従って、ビット配置のそれぞれ異なる生成多項式が合算されて生成された合算生成データを予め記憶する記憶手段と、

ビット列同士の排他的論理和を演算する演算手段と、

ＣＲＣチェック対象のデータを入力する入力手段と、

前記入力手段が入力したデータ又は、前記演算手段により前回演算されて生じた演算結果データから１バイト分のビット配列を特定し、特定したビット配列に対応する合算生成データを前記記憶手段から取得する取得手段と、

前記入力手段が入力したデータ又は、前記演算手段により前回演算されて生じた演算結果データと、前記取得手段が取得した合算生成データとの演算を前記演算手段に所定回数行わせる制御手段と、

前記制御手段の制御により前記演算手段にて所定回数繰り返された演算により生じた演算結果に剰余が含まれる場合に、前記入力手段が入力したデータの誤りを検出するエラー検出手段と、

を備えることを特徴とする誤り検出装置。

【請求項３】ビット列同士の排他的論理和を演算する演算ステップと、

ＣＲＣチェック対象のデータを入力する入力ステップと、

前記入力ステップにて入力されたデータ又は、前記演算ステップにて前回演算されて生じた演算結果データから複数ビット分のビット配列を特定し、特定したビット配列により定まる生成多項式が合算されて生成された合算生成データを所定の記憶部から取得する取得ステップと、

2

前記入力ステップにて入力されたデータ又は、前記演算ステップにて前回演算されて生じた演算結果データと、前記取得ステップにて取得された合算生成データとの演算を前記演算ステップにて所定回数行わせる制御ステップと、

前記制御ステップの制御により前記演算ステップにて所定回数繰り返された演算により生じた演算結果に従って、前記入力ステップにて入力されたデータの誤りを検出するエラー検出ステップと、

を備えることを特徴とするＣＲＣチェック方法。

【請求項４】ビット列同士の排他的論理和を演算する演算ステップと、

ＣＲＣチェック対象のデータを入力する入力ステップと、

前記入力ステップにて入力されたデータ又は、前記演算ステップにて前回演算されて生じた演算結果データから１バイト分のビット配列を特定し、特定したビット配列により定まるビット配置のそれぞれ異なる生成多項式が合算されて生成された合算生成データを所定の記憶部から取得する取得ステップと、

前記入力ステップにて入力されたデータ又は、前記演算ステップにて前回演算されて生じた演算結果データと、前記取得ステップにて取得された合算生成データとの演算を前記演算ステップにて所定回数行わせる制御ステップと、

前記制御ステップの制御により前記演算ステップにて所定回数繰り返された演算により生じた演算結果に剰余が含まれる場合に、前記入力ステップにて入力されたデータの誤りを検出するエラー検出ステップと、

を備えることを特徴とするＣＲＣチェック方法。

【請求項５】ビット列同士の排他的論理和を演算する演算ステップと、ＣＲＣチェック対象のデータを入力する入力ステップと、前記入力ステップにて入力されたデータ又は、前記演算ステップにて前回演算されて生じた演算結果データから複数ビット分のビット配列を特定し、特定したビット配列により定まる生成多項式が合算されて生成された合算生成データを所定の記憶部から取得する取得ステップと、前記入力ステップにて入力されたデータ又は、前記演算ステップにて前回演算されて生じた演算結果データと、前記取得ステップにて取得された合算生成データとの演算を前記演算ステップにて所定回数行わせる制御ステップと、前記制御ステップの制御により前記演算ステップにて所定回数繰り返された演算により生じた演算結果に従って、前記入力ステップにて入力されたデータの誤りを検出するエラー検出ステップとを有するＣＲＣチェック方法をコンピュータに実行させるためのプログラムを記録したコンピュータ読み取り可能な記録媒体。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、取得したデータの誤りをＣＲＣ（Cyclic Redundancy Check）方式により検出する誤り検出装置に関し、特にＣＲＣチェックを高速に処理することのできる誤り検出装置、ＣＲＣチェック方法及び記録媒体に関する。

【０００２】

【従来の技術】従来から、送信されたデータや所定の媒体から読み出したデータ等が誤りなく取得できたか否かを判別するために、データの誤り検出が行われている。この誤り検出には、パリティチェック方式やチェックサム方式が用いられてきた。パリティチェック方式は、例えば、送信側にて、７ないし８ビットで表される１文字分のデータに別の１ビットを付加し、全体の「１」のビット数が常に奇数（又は、偶数）個になるようにしてデータを生成する。そして、受信側にて、取得したデータ（１文字分）の「１」のビット数が、奇数（又は、偶数）個でない場合に、エラーが発生したと判断する。また、チェックサム方式は、例えば、送信側にて、一定の長さのブロック内のデータ列を足し合わせた総和を求め、求めた合計値の最下桁１バイトをブロックの最後尾に付加したデータを生成する。そして、受信側にて、取得したデータのブロック内の合計値を求め、合計値の最下桁１バイトと最後尾の１バイトとを比較して、一致しない場合に、エラーが発生したと判断する。

【０００３】近年では、これらパリティチェック方式等よりもエラーの検出能力が高く、状況によってはエラーの訂正も可能という点等の理由により、ＣＲＣ（Cyclic Redundancy Check）方式が誤り検出に用いられるようになっている。ＣＲＣ方式は、例えば、送信側にて、ブロック単位のデータ毎に所定の生成多項式を使用して巡回符号を求め、求めた巡回符号をそれぞれ付加したデータを生成する。そして、受信側にて、取得したデータをブロック単位に同一の生成多項式にて除算し、割り切れなかった（剰余が生じた）場合に、エラーが発生したと判断する。

【０００４】以下、このＣＲＣ方式について、伝送すべきデータをデータＤ（$D=X^{23}+X^{21}+X^{18}+X^{17}+X^{14}+X^{13}+X^{12}+X^{11}+X^{10}+X^{6}+X^{4}+X^{3}+1$）とし、生成多項式を多項式Ｇ（$G=X^{6}+X^{4}+X^{3}+1$）とした場合を一例として説明する。送信側は、データＤに多項式Ｇの最高次の項であるX^{6}を乗じたデータＤＸ（$DX=X^{29}+X^{27}+X^{24}+X^{23}+X^{20}+X^{19}+X^{18}+X^{17}+X^{16}+X^{12}+X^{10}+X^{9}+X^{6}$）を多項式Ｇで割り、その剰余を巡回符号として求める。すなわち、送信側は、図９に示すように、データＤＸを多項式Ｇで割り、剰余として、$X^{3}+X^{2}+X$を求める。この$X^{3}+X^{2}+X$が巡回符号となる。

【０００５】より具体的に説明すると、この場合、データＤＸは、「１０１００１１００１１１１１０００１０１１００１０００００」と表され、また、多項式Ｇは、「１０１１００１」と表される。そして、データＤＸを多項式Ｇで除算する際において、実際には剰余しか必要としないため（商が不要であるため）、送信側は、図１０に示すように、データＤＸと多項式Ｇとの排他的論理和（ｘｏｒ）を順次演算し、剰余として「１１１０」（すなわち、$X^{3}+X^{2}+X$）を求める。送信側は、この剰余、つまり巡回符号をデータＤＸに付加したデータ（「１０１００１１００１１１１１０００１０１１００１００１１１０」）を生成して受信側に向けて送信する。

【０００６】受信側は、受信したデータを同一の多項式Ｇで除算し、剰余が生じた場合に、エラーが発生したと判断する。すなわち、受信側は、図１１に示すように、受信したデータ（「１０１００１１００１１１１１００００１０１１００１００１１１０」）と多項式Ｇ（「１０１１００１」）との排他的論理和を順次演算して、剰余の有無を判別する。このように、受信したデータと多項式Ｇとを使用した演算により剰余の有無を判別し、受信データの誤り検出を精度良く行うことができる。

【０００７】

【発明が解決しようとする課題】しかし、上述の図１０や図１１を参照して説明したように、多項式Ｇによるデータの除算は、チェック対象データ等の上位ビットが「１」の場合に、ビット配置を合わせた多項式Ｇとの排他的論理和を演算し、また、「０」の場合に、シフトさせて次のビットを参照するという演算処理を繰り返し行う必要がある。すなわち、ビット単位で演算処理を行っており、排他的論理和等の演算を数多く行う必要があるため、誤り検出のための処理負荷が大きくなり、全体の処理効率の低下を招いていた。

【０００８】また、例えば、ＣＤ－ＲＯＭ内に記録したデータ等を読み出す際には、１セクタ分のデータ量である２０６０バイトものデータに対して、誤り検出を行う必要がある。この際、生成多項式には、誤り検出の精度の向上から、より高次の多項式、例えば、多項式$G(x)=X^{32}+X^{31}+X^{16}+X^{15}+X^{4}+X^{3}+X+1$が用いられることとなる。この場合、２０６０バイトものデータのビット毎に排他的論理和等の演算を繰り返し行うため、処理負荷が極めて大きくなり、誤り検出のために長時間に渡って処理を行わなければならないといった問題があった。

【０００９】本発明は、上記実状に鑑みてなされたもので、ＣＲＣチェックを高速に行うことのできる誤り検出装置、ＣＲＣチェック方法及び記録媒体を提供することを目的とする。

【００１０】

【課題を解決するための手段】上記目的を達成するため、本発明の第１の観点に係る誤り検出装置は、複数ビット分のビット配列に従って、所定の生成多項式が合算

されて生成された合算生成データを予め記憶する記憶手段と、ビット列同士の排他的論理和を演算する演算手段と、ＣＲＣ（Cyclic Redundancy Check）チェック対象のデータを入力する入力手段と、前記入力手段が入力したデータ又は、前記演算手段により前回演算されて生じた演算結果データから複数ビット分のビット配列を特定し、特定したビット配列に対応する合算生成データを前記記憶手段から取得する取得手段と、前記入力手段が入力したデータ又は、前記演算手段により前回演算されて生じた演算結果データと、前記取得手段が取得した合算生成データとの演算を前記演算手段に所定回数行わせる制御手段と、前記制御手段の制御により前記演算手段にて所定回数繰り返された演算により生じた演算結果に従って、前記入力手段が入力したデータの誤りを検出するエラー検出手段と、を備えることを特徴とする。

【００１１】この発明によれば、記憶手段は、複数ビット分のビット配列に従って、所定の生成多項式が合算されて生成された合算生成データを予め記憶する。演算手段は、ビット列同士の排他的論理和を演算する。入力手段は、ＣＲＣチェック対象のデータを入力する。取得手段は、入力手段が入力したデータ又は、演算手段により前回演算されて生じた演算結果データから複数ビット分のビット配列を特定し、特定したビット配列に対応する合算生成データを記憶手段から取得する。制御手段は、入力手段が入力したデータ又は、演算手段により前回演算されて生じた演算結果データと、取得手段が取得した合算生成データとの演算を演算手段に所定回数行わせる。エラー検出手段は、制御手段の制御により演算手段にて所定回数繰り返された演算により生じた演算結果に従って、入力手段が入力したデータの誤りを検出する。このように、チェック対象データと合算生成データとの演算を、チェック対象データの複数ビット単位に繰り返すことにより、チェック対象データの生成多項式による除算を少ない演算処理にて実現できる。この結果、ＣＲＣチェックを高速に行うことができる。

【００１２】上記目的を達成するため、本発明の第２の観点に係る誤り検出装置は、１バイト分のビット配列に従って、ビット配置のそれぞれ異なる生成多項式が合算されて生成された合算生成データを予め記憶する記憶手段と、ビット列同士の排他的論理和を演算する演算手段と、ＣＲＣチェック対象のデータを入力する入力手段と、前記入力手段が入力したデータ又は、前記演算手段により前回演算されて生じた演算結果データから１バイト分のビット配列を特定し、特定したビット配列に対応する合算生成データを前記記憶手段から取得する取得手段と、前記入力手段が入力したデータ又は、前記演算手段により前回演算されて生じた演算結果データと、前記取得手段が取得した合算生成データとの演算を前記演算手段に所定回数行わせる制御手段と、前記制御手段の制御により前記演算手段にて所定回数繰り返された演算により生じた演算結果に剰余が含まれる場合に、前記入力手段が入力したデータの誤りを検出するエラー検出手段と、を備えることを特徴とする。

【００１３】この発明によれば、記憶手段は、１バイト分のビット配列に従って、ビット配置のそれぞれ異なる生成多項式が合算されて生成された合算生成データを予め記憶する。演算手段は、ビット列同士の排他的論理和を演算する。入力手段は、ＣＲＣチェック対象のデータを入力する。取得手段は、入力手段が入力したデータ又は、演算手段により前回演算されて生じた演算結果データから１バイト分のビット配列を特定し、特定したビット配列に対応する合算生成データを記憶手段から取得する。制御手段は、入力手段が入力したデータ又は、演算手段により前回演算されて生じた演算結果データと、取得手段が取得した合算生成データとの演算を演算手段に所定回数行わせる。エラー検出手段は、制御手段の制御により演算手段にて所定回数繰り返された演算により生じた演算結果に剰余が含まれる場合に、入力手段が入力したデータの誤りを検出する。このように、チェック対象データと合算生成データとの演算を、チェック対象データの１バイト単位に繰り返すことにより、チェック対象データの生成多項式による除算を少ない演算処理にて実現できる。この結果、ＣＲＣチェックを高速に行うことができる。

【００１４】上記目的を達成するため、本発明の第３の観点に係るＣＲＣチェック方法は、ビット列同士の排他的論理和を演算する演算ステップと、ＣＲＣチェック対象のデータを入力する入力ステップと、前記入力ステップにて入力されたデータ又は、前記演算ステップにて前回演算されて生じた演算結果データから複数ビット分のビット配列を特定し、特定したビット配列により定まる生成多項式が合算されて生成された合算生成データを所定の記憶部から取得する取得ステップと、前記入力ステップにて入力されたデータ又は、前記演算ステップにて前回演算されて生じた演算結果データと、前記取得ステップにて取得された合算生成データとの演算を前記演算ステップにて所定回数行わせる制御ステップと、前記制御ステップの制御により前記演算ステップにて所定回数繰り返された演算により生じた演算結果に従って、前記入力ステップにて入力されたデータの誤りを検出するエラー検出ステップと、を備えることを特徴とする。

【００１５】この発明によれば、演算ステップは、ビット列同士の排他的論理和を演算する。入力ステップは、ＣＲＣチェック対象のデータを入力する。取得ステップは、入力ステップにて入力されたデータ又は、演算ステップにて前回演算されて生じた演算結果データから複数ビット分のビット配列を特定し、特定したビット配列により定まる生成多項式が合算されて生成された合算生成データを所定の記憶部から取得する。制御ステップは、入力ステップにて入力されたデータ又は、演算ステップ

にて前回演算されて生じた演算結果データと、取得ステップにて取得された合算生成データとの演算を演算ステップにて所定回数行わせる。エラー検出ステップは、制御ステップの制御により演算ステップにて所定回数繰り返された演算により生じた演算結果に従って、入力ステップにて入力されたデータの誤りを検出する。このように、チェック対象データと合算生成データとの演算を、チェック対象データの複数ビット単位に繰り返すことにより、チェック対象データの生成多項式による除算を少ない演算処理にて実現できる。この結果、ＣＲＣチェックを高速に行うことができる。

【００１６】上記目的を達成するため、本発明の第４の観点に係るＣＲＣチェック方法は、ビット列同士の排他的論理和を演算する演算ステップと、ＣＲＣチェック対象のデータを入力する入力ステップと、前記入力ステップにて入力されたデータ又は、前記演算ステップにて前回演算されて生じた演算結果データから１バイト分のビット配列を特定し、特定したビット配列により定まるビット配置のそれぞれ異なる生成多項式が合算されて生成された合算生成データを所定の記憶部から取得する取得ステップと、前記入力ステップにて入力されたデータ又は、前記演算ステップにて前回演算されて生じた演算結果データと、前記取得ステップにて取得された合算生成データとの演算を前記演算ステップにて所定回数行わせる制御ステップと、前記制御ステップの制御により前記演算ステップにて所定回数繰り返された演算により生じた演算結果に剰余が含まれる場合に、前記入力ステップにて入力されたデータの誤りを検出するエラー検出ステップと、を備えることを特徴とする。

【００１７】この発明によれば、演算ステップは、ビット列同士の排他的論理和を演算する。入力ステップは、ＣＲＣチェック対象のデータを入力する。取得ステップは、入力ステップにて入力されたデータ又は、演算ステップにて前回演算されて生じた演算結果データから１バイト分のビット配列を特定し、特定したビット配列により定まるビット配置のそれぞれ異なる生成多項式が合算されて生成された合算生成データを所定の記憶部から取得する。制御ステップは、入力ステップにて入力されたデータ又は、演算ステップにて前回演算されて生じた演算結果データと、取得ステップにて取得された合算生成データとの演算を演算ステップにて所定回数行わせる。エラー検出ステップは、制御ステップの制御により演算ステップにて所定回数繰り返された演算により生じた演算結果に剰余が含まれる場合に、入力ステップにて入力されたデータの誤りを検出する。このように、チェック対象データと合算生成データとの演算を、チェック対象データの１バイト単位に繰り返すことにより、チェック対象データの生成多項式による除算を少ない演算処理にて実現できる。この結果、ＣＲＣチェックを高速に行うことができる。

【００１８】上記目的を達成するため、本発明の第５の観点に係る記録媒体は、ビット列同士の排他的論理和を演算する演算ステップと、ＣＲＣチェック対象のデータを入力する入力ステップと、前記入力ステップにて入力されたデータ又は、前記演算ステップにて前回演算されて生じた演算結果データから複数ビット分のビット配列を特定し、特定したビット配列により定まる生成多項式が合算されて生成された合算生成データを所定の記憶部から取得する取得ステップと、前記入力ステップにて入力されたデータ又は、前記演算ステップにて前回演算されて生じた演算結果データと、前記取得ステップにて取得された合算生成データとの演算を前記演算ステップにて所定回数行わせる制御ステップと、前記制御ステップの制御により前記演算ステップにて所定回数繰り返された演算により生じた演算結果に従って、前記入力ステップにて入力されたデータの誤りを検出するエラー検出ステップとを有するＣＲＣチェック方法をコンピュータに実行させるためのプログラムを記録する。

【００１９】

【発明の実施の形態】本発明の実施の形態にかかる誤り検出装置について、以下図面を参照して説明する。

【００２０】図１は、この発明の実施の形態に適用される誤り検出装置の一例を示すブロック図である。図示するように、誤り検出装置１は、データ入力部１１と、演算処理部１２と、記憶部１３と、エラー出力部１４とを備えて構成される。データ入力部１１は、チェック対象となるデータを順次入力し、入力したデータを演算処理部１２に供給する。

【００２１】演算処置部１２は、ＣＰＵ（Central Processing Unit）等からなり、データ入力部１１から供給されたチェック対象データを演算処理することにより、データエラーが生じているか否かを判別する。具体的に演算処理部１２は、後述するＣＲＣチェック処理を実行し、チェック対象データ中の例えば、対象４ビットから定まる合算生成データを使用した演算処理の結果に従ってエラーの有無を判別する。なお、合算生成データについては、後述する。

【００２２】記憶部１３は、ＲＯＭ（Read Only Memory）等からなり、例えば、図２に示すような生成データテーブル１３１を予め記憶する。生成データテーブル１３１は、チェック対象データ中の対象４ビットと、合算生成データとの対応関係を規定するテーブルである。すなわち、生成データテーブル１３１は、１６個の対象４ビット（「００００」～「１１１１」）に対応する合算生成データ（「００００００００００」～「１１１１１０１０１」）を規定するテーブルである。

【００２３】この合算生成データは、排他的論理和の演算に使用される生成多項式を対象４ビットのビット配列に従って合算したデータである。なお、生成多項式には、多項式Ｇ（$G=X^6+X^4+X^3+1$）が用いられ

る。以下、合算生成データについて、具体的に説明する。

【００２４】例えば、チェック対象データにおける対象４ビット（上位４ビット）のビット配列が「１１１１」である場合、図３に示すように、多項式Ｇの先頭位置（ビット配置）をずらしたＬ１、Ｌ２及びＬ４がそれぞれ合算され（排他的論理和が演算され）、ΣＬとして合算生成データが求められる。

【００２５】ところで従来の除算の場合、例えば図４（ａ）に示すように、チェック対象データ（「１１１１・・・」）と多項式Ｇ（「１０１１００１」）との排他的論理和を順次演算する。具体的には、まず対象データとＬ１との排他的論理和を演算し、その演算結果とＬ２との排他的論理和を演算し、そして、更に演算結果とＬ４との排他的論理和を演算する。すなわち、チェック対象データの対象４ビットが「１１１１」の場合、データに対して、以下の数式１に示すような演算が行われる。なお、数式１中に示される（Ｄ）'は、データＤを１回、多項式Ｇで割った結果を示すものとする。

【００２６】

【数１】（Ｄ）'　＝　Ｄ　ｘｏｒ　Ｌ１

（Ｄ）''　＝（Ｄ）'　ｘｏｒ　Ｌ２

（Ｄ）'''＝（Ｄ）''　ｘｏｒ　Ｌ４

【００２７】この数式１から、（Ｄ）'''とＤとの関係を導くと、数式２にて示される関係が成立する。

【００２８】

【数２】

$$(D)''' = ((D \text{ xor } L1) \text{ xor } L2) \text{ XOR } L4 = D \text{ xor } (L1 \text{ xor } L2 \text{ xor } L4)$$

【００２９】数式２に示すように、データＤと、多項式Ｇのビット配置をずらしたＬ１、Ｌ２及びＬ４の合算（排他的論理和）の演算結果との排他的論理和を演算することにより、数式１と同一となる最終的な演算結果を得ることができる。つまり、図４（ｂ）に示すチェック対象のデータと、Ｌ１、Ｌ２及びＬ４が合算されたΣＬとの排他的論理和の演算結果（「００００１１１・・・」）は、図４（ａ）に示す最終的な演算結果と等しくなる。このことから、数式３が成立することとなる。

【００３０】

【数３】（Ｄ）'''＝　Ｄ　ｘｏｒ　ΣＬ

【００３１】すなわち、排他的論理和に使用される多項式Ｇを合算して生成されるΣＬ、すなわち、合算生成データを使用することにより、図４（ａ）に示すような対象４ビット（「１１11」）に対して３回必要であった排他的論理和の演算を、図４（ｂ）に示すように１回にまとめて行うことができる。このような合算生成データが、対象４ビットの１６種類のビット配列（「００００」～「１１１１」）に応じて予め求められ、それぞれのビット配列に対応づけられた生成データテーブル１３１が記憶部１３に記憶されている。

【００３２】図１に戻って、エラー出力部１４は、演算処理部１２の演算処理結果に従って、エラーが検出された場合に、データの誤りを検出したことを示す所定のエラー検出情報を出力する。

【００３３】以下、この誤り検出装置１の動作を、図５を参照して説明する。図５は、誤り検出装置１が行うＣＲＣチェック処理を説明するためのフローチャートである。図５に示すＣＲＣチェック処理は、データ入力部１１にチェック対象のデータが入力された際に、逐次開始される。

【００３４】まず、演算処理部１２は、データ入力部１１から入力データを取得する（ステップＳ１１）。例えば、演算処理部１２は、チェック対象となる３０ビットのデータを取得する。

【００３５】演算処理部１２は、変数ｉに初期値の「１」をセットする（ステップＳ１２）。なお、この変数ｉは、チェック対象データと合算生成データとの排他的論理和の回数をカウントするために使用される。

【００３６】演算処理部１２は、対象４ビットに対応する合算生成データを取得する（ステップＳ１３）。すなわち、演算処理部１２は、チェック対象データ中の対象４ビットを特定し、特定した対象４ビットに対応する合算生成データを記憶部１３に記憶された生成データテーブル１３１から取得する。例えば、演算処理部１２は、図６（ａ）に示すような３０ビットのチェック対象データから対象４ビット（「１０１０」）を特定し、特定した対象４ビットに対応する合算生成データ（「１０１００１００００１」）を図２に示す生成データテーブル１３１から取得する。

【００３７】演算処理部１２は、取得した合算生成データと入力データとの排他的論理和を演算する（ステップＳ１４）。例えば、演算処理部１２は、図６（ａ）に示すように、チェック対象データと合算生成データ（「１０１００１００００１」）との排他的論理和を演算する。

【００３８】演算処理部１２は、変数ｉが「６」より小さいか否かを判別する（ステップＳ１５）。演算処理部１２は、変数ｉが「６」より小さいと判別した場合、変数ｉに「１」を加算して（ステップＳ１６）、上述のステップＳ１３に処理を戻す。そして、演算処理部１２は、前回のステップＳ１４にて演算された結果（演算結果）から次の対象４ビットを特定し、特定した対象４ビットに対応する合算生成データを生成データテーブル１３１から取得する。更に、演算処理部１２は、取得した合算生成データと前回の演算結果との排他的論理和を演算する。このようなステップＳ１３、Ｓ１４の処理を、演算処理部１２は、変数ｉが「６」となるまで繰り返す。

【００３９】すなわち、演算処理部１２は、変数ｉが「２」のときに、図６（ｂ）に示すように対象４ビット（「００１０」）に対応する合算生成データ（「００１０１１００１０」）を生成データテーブル１３１から取得し、演算結果と合算生成データとの排他的論理和を演算する。また、演算処理部１２は、変数ｉが「３」のときに、図６（ｃ）に示すように対象４ビット（「１１１１」）に対応する合算生成データ（「１１１１１１０１０１」）を取得し、排他的論理和を演算する。以下同様に、演算処理部１２は、変数ｉが「４」のときに、図６（ｄ）に示すように排他的論理和を演算し、変数ｉが「５」のときに、図６（ｅ）に示すように排他的論理和を演算し、そして、変数ｉが「６」のときに、図６（ｆ）に示すように排他的論理和を演算する。

【００４０】図５に戻って、演算処理部１２は、ステップＳ１５にて、変数ｉが「６」以上である（「６」より小さくない）と判別した場合、剰余が発生しているか否かを判別する（ステップＳ１７）。すなわち、上述の６回にわたるチェック対象データ（前回の演算結果）と合算生成データとの排他的論理和の演算により得られた演算結果に、剰余が含まれているか否かを判別する。

【００４１】演算処理部１２は、剰余が発生していると判別した場合、所定のエラー検出情報を生成して、エラー出力部１４に供給する（ステップＳ１８）。演算処理部１２は、エラー検出情報をエラー出力部１４に供給すると、ＣＲＣチェック処理を終了する。

【００４２】一方、剰余が発生していないと判別した場合、演算処理部１２は、全データのＣＲＣチェックが完了したか否かを判別する（ステップＳ１９）。すなわち、データ入力部１１が入力したデータを全て処理したか否かを判別する。

【００４３】演算処理部１２は、全データの処理が完了していないと判別した場合、ステップＳ１１に処理を戻し、上述のステップＳ１１～Ｓ１９の処理を繰り返し実行する。一方、全データの処理が完了したと判別した場合、演算処理部１２は、ＣＲＣチェック処理を終了する。

【００４４】このように、チェック対象データと合算生成データとの演算を、チェック対象データの４ビット単位に繰り返すことにより、チェック対象データの生成多項式による除算を少ない演算処理にて実現できる。この結果、ＣＲＣチェックを高速に行うことができる。

【００４５】次に、上述した誤り検出装置１を構成に含んだ具体的な機器について、ＣＤ－ＲＯＭ装置を一例として説明する。図７は、本発明の他の実施の形態に係るＣＤ－ＲＯＭ装置の一例を示す模式図である。図示するように、ＣＤ－ＲＯＭ装置１００は、誤り検出装置１と、処理制御部２と、制御信号入力部３と、信号処理回路４と、トラッキングスレッドサーボ回路５と、光ピックアップ６と、スピンドルサーボ回路７と、データバッファ８とを備えて構成される。

【００４６】誤り検出装置１は、上述の図１に示したように、データ入力部１１と、演算処理部１２と、記憶部１３と、エラー出力部１４とからなり、信号処理回路４から供給された例えば、２０６０バイトのデータを演算処理し、データエラーが生じているか否かを判別する。

【００４７】なお、記憶部１３には、上述の図２と異なり、図８（ａ）に示すような生成データテーブル１３２が予め記憶される。この生成データテーブル１３２は、チェック対象データ中の対象１バイトと、合算生成データとの対応関係を規定するテーブルである。すなわち、生成データテーブル１３２は、２５６個の対象１バイト（「００００００００」～「１１１１１１１１」）に対応する合算生成データを規定するテーブルである。

【００４８】この合算生成データは、排他的論理和の演算に使用される生成多項式を対象１バイトのビット配列に従って合算したデータである。なお、生成多項式には、多項式Ｇ（ｘ）＝$X^{32}+X^{31}+X^{16}+X^{15}+X^{4}+X^{3}+X+1$が用いられる。これは、チェック対象のデータが２０６０バイトであることから、誤り検出の精度を向上させるためである。また、対象を１バイトとしたのは、ＣＲＣチェック処理をバイト単位に行うことにより、処理効率を更に高めるためである。

【００４９】合算生成データについて具体的に説明すると、合算生成データは、例えば、チェック対象データにおける対象１バイト（上位１バイト）のビット配列が「１１１１１１１１」である場合、図８（ｂ）に示すように、多項式Ｇ（ｘ）の先頭位置（ビット配置）をずらしたＬ１、Ｌ３、Ｌ５及びＬ７がそれぞれ合算され（排他的論理和が演算され）、ΣＬとして求められる。同様に他の合算生成データも、対象１バイトのビット配列に応じて予め求められる。このような、合算生成データが、対象１バイトの２５６種類のビット配列に応じて予め求められ、それぞれのビット配列に対応づけられた生成データテーブル１３２が記憶部１３に記憶されている。

【００５０】図７に戻って、処理制御部２は、ＣＰＵ及び周辺ＬＳＩ（Large Scale Integration）を含んだ１チップマイコン等からなり、ＣＤ－ＲＯＭ装置１００全体を制御する。すなわち、処理制御部２は、信号処理回路４を制御して、信号処理回路４が図示せぬＣＤ－ＲＯＭから光ピックアップ６を介して読み出したデータを誤り検出装置１に供給させる。また、処理制御部２は、誤り検出装置１からエラー検出情報を取得すると、信号処理回路４に対してデータの再読み出しを指示する。

【００５１】制御信号入力部３は、所定のパーソナルコンピュータ等の機器から送られる制御信号を入力し、処理制御部２に供給する。

【００５２】信号処理回路４は、トラッキングスレッドサーボ回路５、光ピックアップ６及び、スピンドルサーボ回路７等を制御する。そして、信号処理回路４は、光

ピックアップ６を介して、ＣＤ－ＲＯＭに記録されたデータを読み出す。例えば、信号処理回路４は、１セクタ分となる２０６０バイトのデータをＣＤ－ＲＯＭから読み出し、読み出した情報を誤り検出装置１等に供給する。

【００５３】トラッキングスレッドサーボ回路５は、光ピックアップ６をＣＤ－ＲＯＭの径方向に平行移動させるための図示せぬスレッドモータを駆動制御し、光ピックアップ６をＣＤ－ＲＯＭの記録面上の所定の位置へ移動させる。

【００５４】光ピックアップ６は、所定の波長のレーザ光をＣＤ－ＲＯＭの記録面に向けて照射し、その反射光を受けて電気信号に変換する。光ピックアップ６は、変換した電気信号を読み出したデータとして信号処理回路４に供給する。

【００５５】スピンドルサーボ回路７は、ＣＤ－ＲＯＭを搭載する所定のターンテーブルを回転させるための図示せぬスピンドルモータを駆動制御し、所定の回転速度で回転駆動させる。

【００５６】データバッファ８は、誤り検出装置１にてＣＲＣチェックがなされたデータを信号処理回路４から取得し、所定のパーソナルコンピュータ等に向けて出力する。

【００５７】以下、上述のＣＤ－ＲＯＭ装置１００における誤り検出装置１の動作を、簡単に説明する。誤り検出装置１は、信号処理回路４から送られる２０６０バイトのチェック対象データを取得する。誤り検出装置１は、チェック対象データ中の対象１バイトを特定し、特定した１バイトのビット配列に対応する合算生成データを生成データテーブル１３２から取得する。誤り検出装置１は、取得した合算生成データとチェック対象データとの排他的論理和を演算する。そして、誤り検出装置１は、前回の演算結果から次の対象１バイトを特定し、特定した対象１バイトのビット配列に対応する合算生成データを生成データテーブル１３２から取得する。更に、演算処理部１２は、取得した合算生成データと前回の演算結果との排他的論理和を演算する。このような演算処理を所定回数繰り返した後に、誤り検出装置１は、最終的に得られた演算結果に、剰余が含まれているか否かを判別する。誤り検出装置１は、剰余が含まれていると判別した場合、処理制御部２にエラー検出情報を出力する。

【００５８】このように、チェック対象データと合算生成データとの演算を、チェック対象データの１バイト単位に繰り返すことにより、チェック対象データの生成多項式による除算を少ない演算処理にて実現できる。この結果、ＣＲＣチェックを高速に行うことができる。

【００５９】なお、上記の実施の形態では、ＣＤ－ＲＯＭ装置等におけるＣＲＣチェックについて説明したが、ＣＲＣチェックの対象は任意である。例えば、ＦＤドライブ、メモリ、ＭＰＥＧ変換装置、通信装置、及び、ハードディスク等種々の機器にて行われるＣＲＣチェックに適用可能である。

【００６０】なお、この発明の誤り検出装置は、専用のシステムによらず、通常のコンピュータシステムを用いて実現可能である。例えば、コンピュータに上述のいずれかを実行するためのプログラムを格納した媒体（フロッピーディスク、ＣＤ－ＲＯＭ等）から該プログラムをインストールすることにより、上述の処理を実行する誤り検出装置を構成することができる。

【００６１】また、コンピュータにプログラムを供給するための手法は、任意である。例えば、通信回線、通信ネットワーク、通信システム等を介して供給してもよい。一例を挙げると、通信ネットワークの掲示板（ＢＢＳ）に当該プログラムを掲示し、これをネットワークを介して配信する。そして、このプログラムを起動し、ＯＳの制御下で、他のアプリケーションプログラムと同様に実行することにより、上述の処理を実行することができる。

【００６２】

【発明の効果】以上説明したように、本発明によれば、ＣＲＣチェックを高速に行うことができる。

【図面の簡単な説明】

【図１】本発明の実施の形態に係る誤り検出装置の構成の一例を示すブロック図である。

【図２】記憶部に記憶される生成データテーブルの一例を示す模式図である。

【図３】合算生成データの算出について説明する模式図である。

【図４】（ａ）が従来の多項式を使用して排他的論理和の演算を順次行う様子を説明するための模式図であり、（ｂ）が合算生成データを使用して排他的論理和の演算を行う様子を説明するための模式図である。

【図５】本発明の実施の形態に係るＣＲＣチェック処理を説明するためのフローチャートである。

【図６】ＣＲＣチェック処理により、チェック対象データと合算生成データとの排他的論理和の演算が順次行われる様子を説明するための模式図である。

【図７】誤り検出装置が含まれるＣＤ－ＲＯＭ装置の構成の一例を説明するための模式図である。

【図８】（ａ）が記憶部に記憶される生成データテーブルの一例を示す模式図であり、（ｂ）が合算生成データの算出について説明する模式図である。

【図９】生成多項式にてデータが除算される様子を説明するための模式図である。

【図１０】従来の巡回符号生成処理において演算される対象データと多項式との排他的論理和について説明するための模式図である。

【図１１】従来のＣＲＣチェック処理において演算される対象データと多項式との排他的論理和について説明す

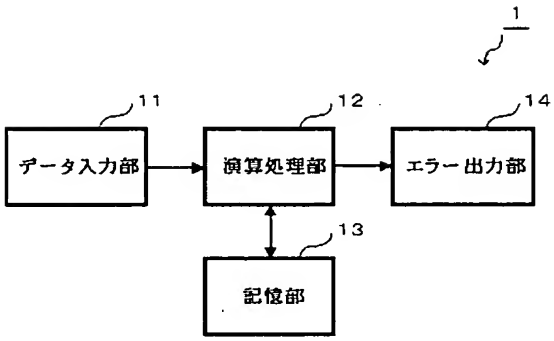
15　　16

るための模式図である。

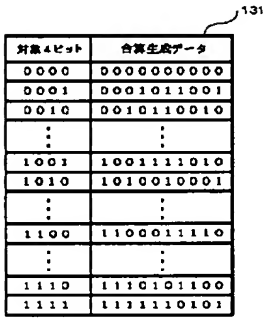
【符号の説明】

1　誤り検出装置
2　処理制御部
3　制御信号入力部
4　信号処理回路
5　トラッキングスレッドサーボ回路
6　光ピックアップ
7　スピンドルサーボ回路
8　データバッファ
１１　データ入力部
１２　演算処理部
１３　記憶部
１４　エラー出力部

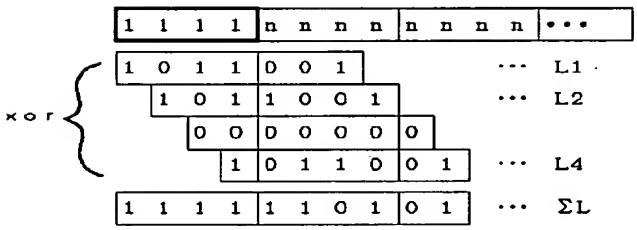
【図１】



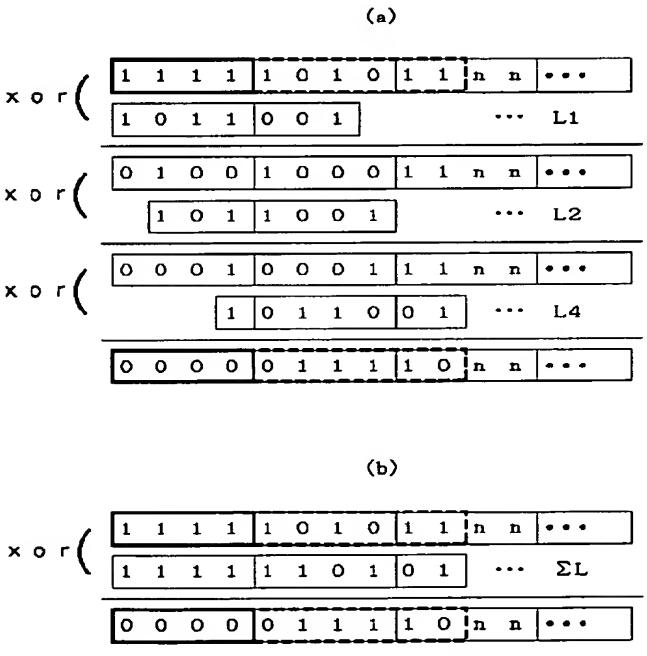
【図２】



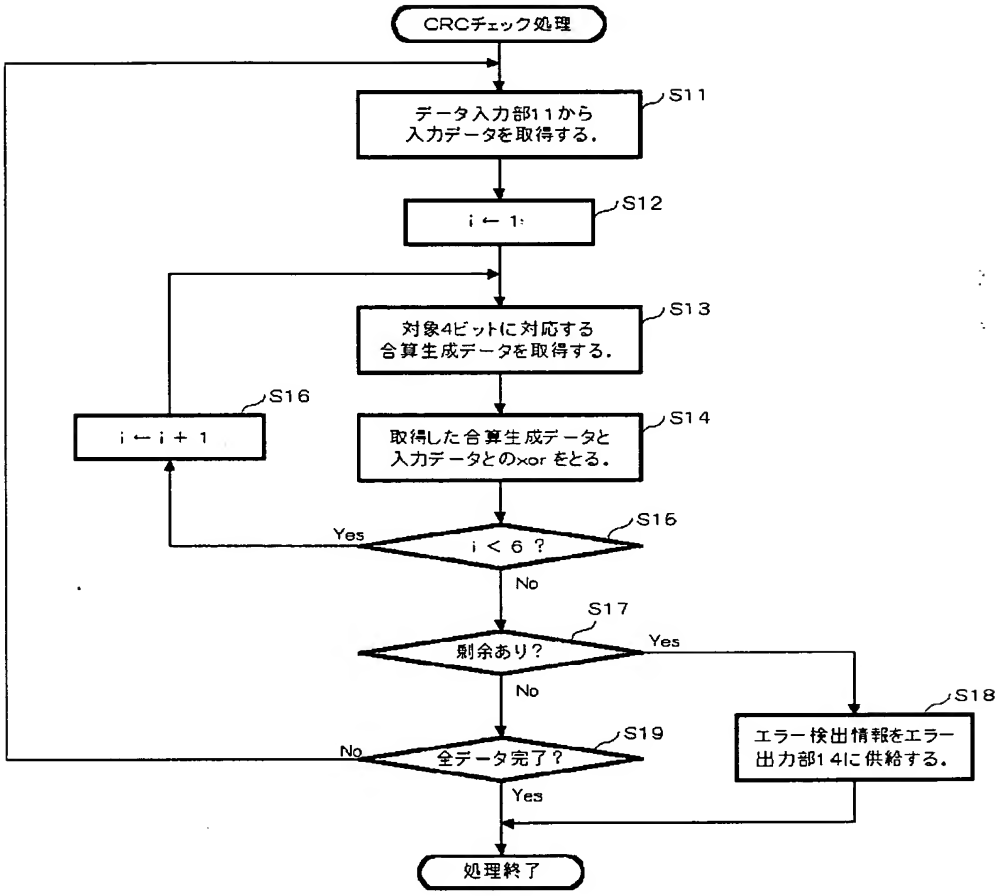
【図３】



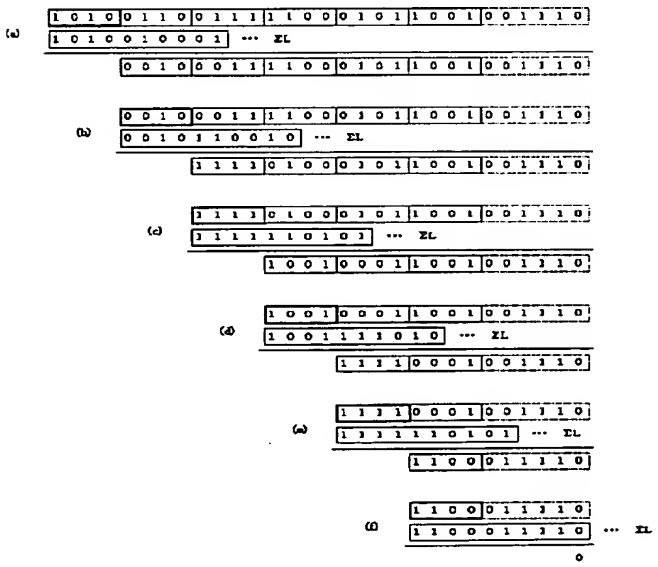
【図４】



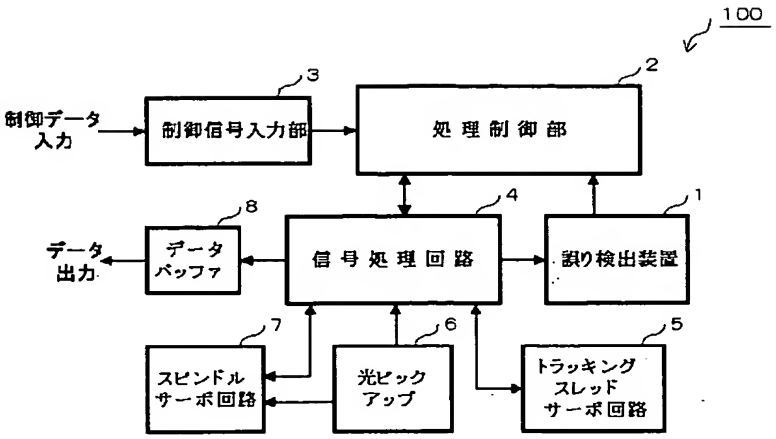
【図５】


CRCチェック処理
データ入力部11から
入力データを取得する。
S11
i ← 1
S12
対象4ビットに対応する
合算生成データを取得する。
S13
取得した合算生成データと
入力データとのxor をとる。
S14
i ← i + 1
S16
i < 6 ?
S15
Yes
No
剰余あり?
S17
Yes
No
エラー検出情報をエラー
出力部14に供給する。
S18
全データ完了?
S19
No
Yes
処理終了

【図６】



【図７】



【図８】

(a)

132

対象1バイト	合算生成データ
00000000	00000000000000000000000000000000000000000000
⋮	⋮
11111111	11111111000000001111111100001110000011 10

(b)

11111111	D2059(byte)	D2058(byte)	D2057(byte)	D2056(byte)	・・・
11000000	00000000	11000000	00001101	1	・・・ L1
00000000	00000000	00000000	00000000	00	
11000000	00000000	00110000	00000011	011	・・・ L3
00000000	00000000	00000000	00000000	0000	
11000000	00000000	00001100	00000000	11011	・・・ L5
00000000	00000000	00000000	00000000	000000	
11000000	00000000	00000011	00000000	001101 1	・・・ L7
00000000	00000000	00000000	00000000	00000000	
11111111	00000000	11111111	00001110	00001110	・・・ ΣL

xor

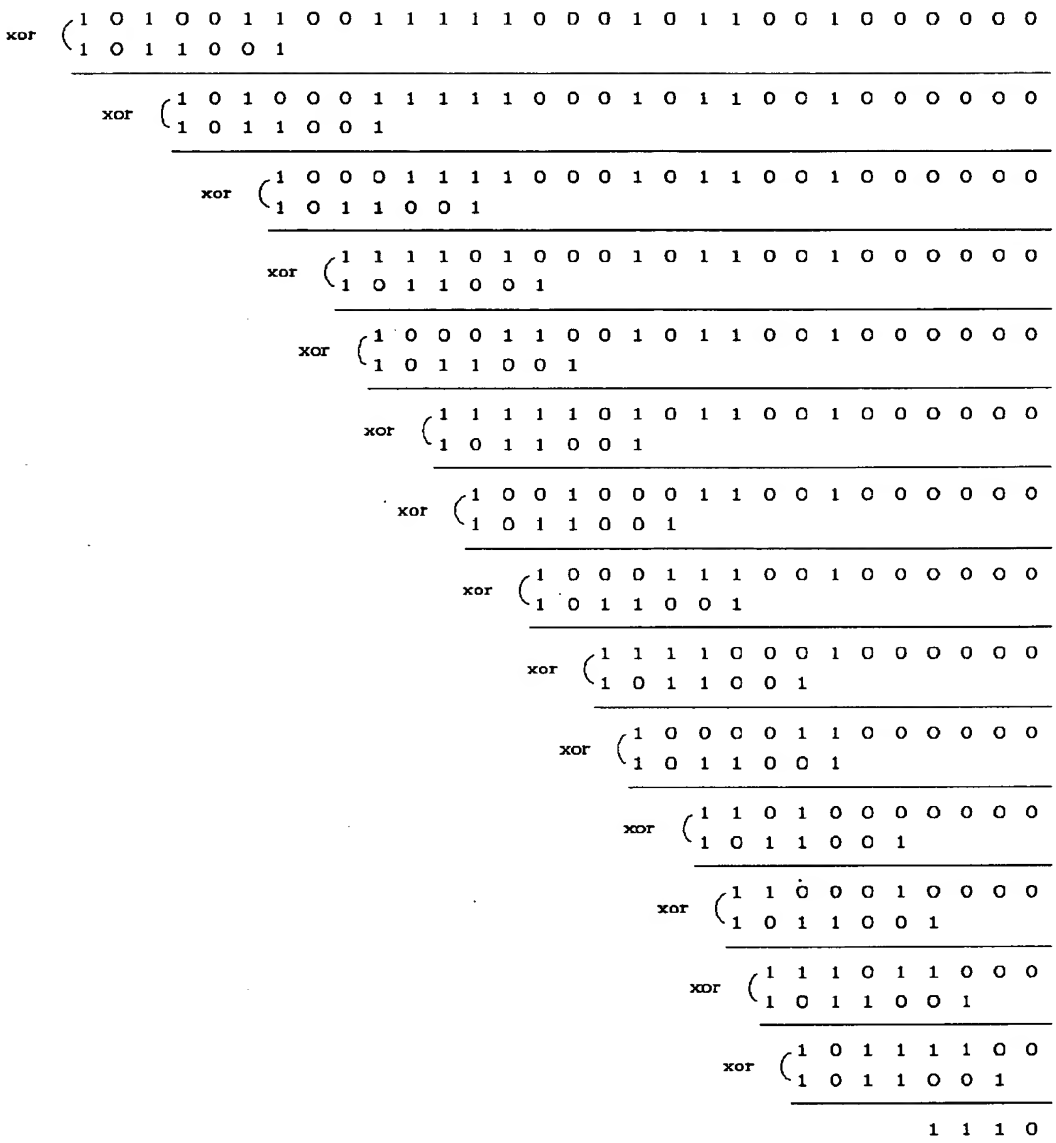
【図９】

$$
\begin{array}{r}
x^{23}+x^{20}+x^{17}+x^{15}+x^{14}+x^{12}+x^{11}+x^{9}+x^{7}+x^{6}+x^{3}+x^{2}+x \\
\hline
x^{6}+x^{4}+x^{3}+1 \;\big)\; x^{29}+x^{27}+x^{24}+x^{23}+x^{20}+x^{19}+x^{18}+x^{17}+x^{16}+x^{12}+x^{10}+x^{9}+x^{6} \\
x^{29}+x^{27}+x^{26}+x^{23} \\
\hline
x^{26}+x^{24}+x^{20}+x^{19}+x^{18}+x^{17}+x^{16}+x^{12}+x^{10}+x^{9}+x^{6} \\
x^{26}+x^{24}+x^{23}+x^{20} \\
\hline
\vdots \\
\hline
x^{7}+x^{5}+x^{4}+x^{3}+x^{2} \\
x^{7}+x^{5}+x^{4}+x \\
\hline
x^{3}+x^{2}+x
\end{array}
$$

【図１１】

$$
\begin{array}{l}
\text{xor}\ \begin{pmatrix}1&0&1&0&0&1&1&0&0&1&1&1&1&1&0&0&0&1&0&1&1&0&0&1&0&0&1&1&1&0\\ 1&0&1&1&0&0&1\end{pmatrix} \\
\hline
\quad \text{xor}\ \begin{pmatrix}1&0&1&0&0&0&1&1&1&1&1&0&0&0&1&0&1&1&0&0&1&0&0&1&1&1&0\\ 1&0&1&1&0&0&1\end{pmatrix} \\
\hline
\qquad \vdots \\
\hline
\qquad \text{xor}\ \begin{pmatrix}1&1&1&0&1&0&1&1&0\\ 1&0&1&1&0&0&1\end{pmatrix} \\
\hline
\qquad\quad \text{xor}\ \begin{pmatrix}1&0&1&1&0&0&1&0\\ 1&0&1&1&0&0&1\end{pmatrix} \\
\hline
\qquad\qquad\qquad 0
\end{array}
$$

【図１０】


xor (1 0 1 0 0 1 1 0 0 1 1 1 1 1 0 0 0 1 0 1 1 0 0 1 0 0 0 0 0 0 0
1 0 1 1 0 0 1
xor (1 0 1 0 0 0 1 1 1 1 1 0 0 0 1 0 1 1 0 0 1 0 0 0 0 0 0
1 0 1 1 0 0 1
xor (1 0 0 0 1 1 1 1 0 0 0 1 0 1 1 0 0 1 0 0 0 0 0 0
1 0 1 1 0 0 1
xor (1 1 1 1 0 1 0 0 0 1 0 1 1 0 0 1 0 0 0 0 0 0
1 0 1 1 0 0 1
xor (1 0 0 0 1 1 0 0 1 0 1 1 0 0 1 0 0 0 0 0 0
1 0 1 1 0 0 1
xor (1 1 1 1 1 0 1 0 1 1 0 0 1 0 0 0 0 0 0
1 0 1 1 0 0 1
xor (1 0 0 1 0 0 0 1 1 0 0 1 0 0 0 0 0 0
1 0 1 1 0 0 1
xor (1 0 0 0 1 1 1 0 0 1 0 0 0 0 0 0
1 0 1 1 0 0 1
xor (1 1 1 1 0 0 0 1 0 0 0 0 0 0
1 0 1 1 0 0 1
xor (1 0 0 0 0 1 1 0 0 0 0 0 0
1 0 1 1 0 0 1
xor (1 1 0 1 0 0 0 0 0 0 0
1 0 1 1 0 0 1
xor (1 1 0 0 0 1 0 0 0 0
1 0 1 1 0 0 1
xor (1 1 1 0 1 1 0 0 0
1 0 1 1 0 0 1
xor (1 0 1 1 1 1 0 0
1 0 1 1 0 0 1
1 1 1 0